✓ 67531-US
KK/mK

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００２年１２月１３日

出願番号
Application Number: 特願２００２－３６２４８７

[ST. 10/C]: [ＪＰ２００２－３６２４８７]

出願人
Applicant(s): 株式会社デンソー

２００３年 ８月１８日

特許庁長官
Commissioner,
Japan Patent Office 今井康夫

出証番号 出証特２００３－３０６７２３０

【書類名】　特許願
【整理番号】　PNID3972
【提出日】　平成14年12月13日
【あて先】　特許庁長官殿
【国際特許分類】　G06F 15/00
【発明者】
　【住所又は居所】　愛知県刈谷市昭和町１丁目１番地　株式会社デンソー内
　【氏名】　宮本　誠司
【発明者】
　【住所又は居所】　愛知県刈谷市昭和町１丁目１番地　株式会社デンソー内
　【氏名】　井原　博之
【特許出願人】
　【識別番号】　000004260
　【氏名又は名称】　株式会社デンソー
【代理人】
　【識別番号】　100082500
　【弁理士】
　【氏名又は名称】　足立　勉
　【電話番号】　052-231-7835
【手数料の表示】
　【予納台帳番号】　007102
　【納付金額】　21,000円
【提出物件の目録】
　【物件名】　明細書　1
　【物件名】　図面　1
　【物件名】　要約書　1
　【包括委任状番号】　9004766
【プルーフの要否】　要

【書類名】明細書

【発明の名称】車両用制御プログラム、車両用制御装置、車両用制御プログラムの生成方法

【特許請求の範囲】

【請求項１】

プラットフォームプログラムと、アプリケーションプログラムとを備える車両用制御プログラムにおいて、さらに結合処理プログラムを備え、

前記プラットフォームプログラムは、ハードウェアデバイスからデータを入力し、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースであるＰＦインターフェースに従って、前記入力したデータに基づくデータを前記結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムであり、

前記結合処理プログラムは、前記ＰＦインターフェースに従って前記プラットフォームプログラムによる処理から提供されたデータを、開発対象の車両用制御装置の要求仕様を満たすインターフェースであるＡＰインターフェースに適合するように変換することで前記アプリケーションプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムであり、

前記アプリケーションプログラムは、前記ＡＰインターフェースに従って前記結合処理プログラムによる処理から提供されたデータを用いた処理をコンピュータに実行させるためのプログラムであること

を特徴とする車両用制御プログラム。

【請求項２】

請求項１に記載の車両用制御プログラムにおいて、

前記ＰＦインターフェースとして標準化する対象は、当該ＰＦインターフェースを介して提供するデータの構造及び当該データの更新タイミングであること

を特徴とする車両用制御プログラム。

【請求項３】

請求項１または２に記載の車両用制御プログラムにおいて、

前記プラットフォームプログラムは、前記ＰＦインターフェースを介して提供するデータを、前記アプリケーションプログラムによる処理で要求される精度よりも高い精度のデータ形式で提供する処理をコンピュータに実行させるためのプログラムを備え、

前記結合処理プログラムは、前記アプリケーションプログラムの要求仕様を満たすデータ形式となるように、前記ＰＦインターフェースを介して提供されたデータのデータ形式を調整して前記ＡＰインターフェースに従って提供する処理をコンピュータに実行させるためのプログラムを備えること

を特徴とする車両用制御プログラム。

【請求項４】

請求項１～３のいずれかに記載の車両用制御プログラムにおいて、

前記プラットフォームプログラムは、前記ＰＦインターフェースに従って提供するデータを、前記アプリケーションプログラムによる処理で要求されるよりも高いサンプリングタイミングで提供する処理をコンピュータに実行させるためのプログラムを備え、

前記結合処理プログラムは、前記要求仕様を満たすデータのサンプリングタイミングへ、前記ＰＦインターフェースを介して提供されたデータの提供タイミングを調整し、前記ＡＰインターフェースによって提供する処理をコンピュータに実行させるためのプログラムを備えること

を特徴とする車両用制御プログラム。

【請求項５】

請求項１～４のいずれかに記載の車両用制御プログラムにおいて、

前記ＡＰインターフェースは、前記変換したデータを、前記アプリケーションプログラムによる処理によって参照可能に構成したものであること

を特徴とする車両用制御プログラム。

【請求項６】

請求項１～５のいずれかに記載の車両用制御プログラムにおいて、

前記ＡＰインターフェースは、前記アプリケーションプログラムによる処理によってデータの要求があった場合に、前記変換したデータを前記アプリケーショ

ンプログラムによる処理に渡すように構成したものであること

を特徴とする車両用制御プログラム。

【請求項７】

請求項１～６のいずれかに記載の車両用制御プログラムにおいて、

前記結合処理プログラムは、前記ＰＦインターフェースを介して取得したデータまたは前記変換したデータを記憶手段に記憶しておき、当該記憶手段に記憶されたデータを変換したデータあるいは当該記憶手段に記憶されたデータを前記ＡＰインターフェースを介して提供するプログラムを備えること

を特徴とする車両用制御プログラム。

【請求項８】

請求項７に記載の車両用制御プログラムにおいて、

前記プラットフォームプログラムは、前記ハードウェアデバイスからの割り込みによって、前記ハードウェアデバイスからのデータの入力を前記コンピュータに行わせるための割り込み処理プログラムを備え、前記割り込み処理プログラムは、当該入力結果に基づくデータを前記ＰＦインターフェースを介して提供する処理をコンピュータに実行させるためのプログラムを備え、

前記結合処理プログラムは、さらに、他の処理のディスパッチを禁止している間に、前記ＰＦインターフェースを介してデータを受け取り、前記記憶手段に記憶する処理をコンピュータに実行させるためのプログラムである取得プログラムを備えること

を特徴とする車両用制御プログラム。

【請求項９】

プラットフォームプログラムと、アプリケーションプログラムとを備える車両用制御プログラムにおいて、さらに結合処理プログラムを備え、

前記プラットフォームプログラムは、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースであるＰＦインターフェースに従って出力のためのデータを前記結合処理プログラムによる処理から取得し、取得したデータに基づくデータを出力する処理をコンピュータに実行させるためのプログラムで

あり、

前記結合処理プログラムは、開発対象の車両用制御装置の要求仕様を満たすインターフェースであるＡＰインターフェースに従って前記アプリケーションプログラムによる処理から提供されたデータを、前記ＰＦインターフェースに適合するように変換して、前記プラットフォームプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムであり、

前記アプリケーションプログラムは、出力対象のデータを生成し、前記ＡＰインターフェースに従って前記出力対象のデータを前記結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムであること

を特徴とする車両用制御プログラム。

【請求項１０】

請求項９に記載の車両用制御プログラムにおいて、

前記ＡＰインターフェースは、前記アプリケーションプログラムによる処理によって記憶されたデータを結合処理プログラムによる処理によって自ら取得する構成とすること

を特徴とする車両用制御プログラム。

【請求項１１】

請求項１～８のいずれかに記載の車両用制御プログラムと請求項９または１０に記載の車両用制御プログラムとを備える車両用制御プログラム。

【請求項１２】

請求項１～１１のいずれかに記載の車両用制御プログラムと当該車両用制御プログラムを実行するコンピュータとを備えた車両用制御装置。

【請求項１３】

ハードウェアデバイスからデータを入力し、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースであるＰＦインターフェースに従って、前記入力したデータに基づくデータを前記結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムであるプラットフォームプログラムを生成し、

前記ＰＦインターフェースに従って前記プラットフォームプログラムによる処理から提供されたデータを、開発対象の車両用制御装置の要求仕様を満たすインターフェースであるＡＰインターフェースに適合するように変換することで前記アプリケーションプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムである結合処理プログラムを生成し、

前記ＡＰインターフェースに従って前記結合処理プログラムによる処理から提供されたデータを用いた処理をコンピュータに実行させるためのプログラムであるアプリケーションプログラムを生成し、

前記プラットフォームプログラムと、前記結合処理プログラムと、前記アプリケーションプログラムとをリンクして車両用制御プログラムを生成すること

を特徴とする車両用制御プログラムの生成方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

車両用制御プログラム、例えばエンジン制御用組込ソフトウェアとして好適な構成をもつプログラム等に関する。

【０００２】

【従来の技術】

従来より、例えばマイコン等のコンピュータを用いて、自動車等の車両の各部を制御する車両用制御装置が知られている。こうした車両用制御装置では、ＲＯＭ等の記憶装置に記憶された車両用制御プログラムをコンピュータが実行することにより、種々の制御を実現している。

【０００３】

車両用制御プログラムは、一般的に、ハードウェアに依存するプログラムであるプラットフォームプログラム（ＰＦ）と、車両用の制御に依存するプログラムであるアプリケーションプログラム（ＡＰ）とに分けて構成する。

プラットフォームプログラムは、ハードウェアデバイスからデータを入力して、その入力したデータに基づくデータを、例えばＡＰＩ（Application Program Interface）のようなインターフェース（以下このインターフェースをＰＦイン

ターフェースと称する）を介して、アプリケーションプログラムによる処理に対して提供したり、アプリケーションプログラムによる処理からこのＰＦインターフェースを介して得たデータをハードウェアデバイスに対して出力したりする処理をコンピュータに実行させるためのプログラムである。

【０００４】

一方、アプリケーションプログラムは、このＰＦインターフェースを介して入力したデータを用いた処理や、ＰＦインターフェースを介して出力するデータを生成する処理をコンピュータに実行させるためのプログラムである。例えばＰＦインターフェースを介して入力したデータに基づいてＰＦインターフェースを介して出力するデータを生成する処理をコンピュータに実行させるためのプログラムである。

【０００５】

このように、プラットフォームプログラムは、ハードウェアデバイスに対するアクセスを行うプログラムを含む一方、アプリケーションプログラムは、直接ハードウェアデバイスへのアクセスを行うプログラムを含まず、ＰＦインターフェースを介して、プラットフォームプログラムによる処理にハードウェアデバイスとの入出力をさせている。こうすることで、アプリケーションプログラムの開発者は、ハードウェアの詳細構成や制御方法を意識することなくアプリケーションプログラムを作成することができる。

【０００６】

また、従来、例えば、特許文献１に示すように、オペレーティングシステムとアプリケーションソフトとを接続するインターフェースソフトを設け、アプリケーションソフトにおいて必要な信号の演算をインターフェースソフトで行うようにすることで、ハードウェアに変更があった場合にインターフェースソフトを書き換えるだけで「アプリケーションソフトを」永続的に使用可能にした自動車用制御ユニットも知られている。

【０００７】

【特許文献１】

特開平０７－０４０７９４号公報

【０００８】

【発明が解決しようとする課題】

しかしながら、車両用制御装置のメーカでは、顧客である車両メーカから異なる要求仕様を持つ複数の車両用制御装置の開発を依頼されることが多く、アプリケーションプログラムはそれぞれの要求仕様に応じて新規に作成する必要がある。したがって、上述した特許文献１に記載のように「アプリケーションプログラムソフトを」流用して車両用制御プログラムを作成するのではなく、プラットフォームプログラムを、できるだけ流用して、車両用制御プログラムを構成することにより、開発コストを削減する手法が採られる。

【０００９】

プラットフォームプログラムを流用するためには、例えば、異なる要求仕様間の共通項を抽出し、その共通項で標準化したＰＦインターフェースを提供するようにプラットフォームプログラムを構成する。例えば各要求仕様の中で、最も要求の厳しい要求仕様に合わせたデータ型式やデータ提供タイミング等でデータをアプリケーションプログラムによる処理へ提供するようにＰＦインターフェースの仕様であるＰＦインターフェース仕様を標準化する。

【００１０】

例えば車両用制御装置がエンジン制御装置であって、エンジンの冷却水の水温を検出するための水温センサからの電圧値に基づく処理を行う場合の要求仕様として、ある車種のエンジン制御装置Ａでは、この電圧値の分解能として1/256Vが要求仕様であるのに対し、別の車種のエンジン制御装置Ｂでは分解能として1/128Vが要求仕様とされている場合を考える。このような場合には、プラットフォームプログラムによる処理では両方の車種のエンジン制御装置の要求仕様を共に満たすように分解能の細かいエンジン制御装置Ａに合わせた分解能1/256Vで、アプリケーションプログラムによる処理に対してＰＦインターフェースを提供するＰＦインターフェース仕様を策定し、このＰＦインターフェース仕様に沿ったＰＦインターフェースを提供するようにプラットフォームプログラムを構成する。こうして、異なる要求仕様を持つエンジン制御装置で同一のプラットフォームプログラムを流用することができる。

【００１１】

しかしながら、エンジン制御装置Ｂのアプリケーションプログラムを作成する場合、要求仕様の分解能は1/128Vであるのに対し、プラットフォームプログラムによるＰＦインターフェースが提供する分解能は1/256Vであるため、アプリケーションプログラムの開発者は、プラットフォームプログラムによるＰＦインターフェース仕様の分解能と、開発対象の車両用制御装置の要求仕様の分解能とのギャップを埋めるための分解能変換処理プログラムをアプリケーションプログラムに組み込む必要がある。

【００１２】

こうした要求仕様としては、分解能の他にも、データの種類、データの型、データの取り込みタイミング等、様々なものがあり、異なる要求仕様を持つ車両用制御装置において標準化されたプラットフォームプログラムを利用するアプリケーションプログラムを作成する場合、アプリケーションプログラムの開発者は、常にプラットフォームプログラムの提供するＰＦインターフェース仕様を念頭において要求仕様を実現する方法を考える必要があり、アプリケーションプログラムの開発者の負担が大きいといった問題がある。

【００１３】

そこで本発明は、アプリケーションプログラムの開発者の負担を軽減することのできる車両用制御プログラム等を提供することを目的とする。

【００１４】

【課題を解決するための手段及び発明の効果】

上述した問題点を解決するためになされた請求項１に記載の車両用制御プログラムは、プラットフォームプログラムと、アプリケーションプログラムと、結合処理プログラムとから構成されている。

【００１５】

そして、プラットフォームプログラムは、ハードウェアデバイスからデータを入力し、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースであるＰＦインターフェースに従って、前記入力したデータに基づくデータ

を結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムである。

【００１６】

また、結合処理プログラムは、ＰＦインターフェースに従ってプラットフォームプログラムによる処理から提供されたデータを、開発対象の車両用制御装置の要求仕様を満たすインターフェースであるＡＰインターフェースに適合するように変換することでアプリケーションプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムである。

【００１７】

そして、アプリケーションプログラムは、ＡＰインターフェースに従って結合処理プログラムによる処理から提供されたデータを用いた処理をコンピュータに実行させるためのプログラムである。

このように、結合処理プログラムが、ＰＦインターフェースと、ＡＰインターフェースとの橋渡しを行うため、アプリケーションプログラムの開発者は、ＡＰインターフェース仕様、すなわち、開発対象の車両用制御装置の要求仕様を満たすインターフェース仕様に沿ってアプリケーションプログラムを作成することができる。すなわち、アプリケーションプログラムの開発者は、ＰＦインターフェース仕様を念頭において要求仕様を実現する方法を考える必要はなくなるので、アプリケーションプログラムの開発者の負担を軽減することができる。

【００１８】

このようなＰＦインターフェースとして標準化する対象としては、例えば、請求項２に示すように、ＰＦインターフェースを介して提供するデータの構造及び当該データの更新タイミングなどが挙げられる。異なる要求仕様に従って作成されたアプリケーションプログラムは、異なるデータの構造を持つデータを必要とし、データの必要となるタイミング等も異なる。例えば、データの構造としては、どの入力対象（例えばどのセンサ）からの、どのような精度の、どのようなデータ型式（データ構造）のデータなのかという事項などを標準化して提供する。

【００１９】

また、例えば、請求項３に示す車両用制御プログラムによれば、プラットフォ

ームプログラムによる処理で、ＰＦインターフェースとして標準化して提供するデータを、アプリケーションプログラムによる処理で要求される精度よりも高い精度のデータ形式で提供し、結合処理プログラムによる処理で、アプリケーションプログラムの要求仕様を満たすデータ型式となるように、ＰＦインターフェースを介して提供されたデータのデータ型式を調整してＡＰインターフェースによって提供する。このようにすれば、アプリケーションプログラムの作成時にＰＦインターフェースにおけるデータがどのような精度やデータ型式となっているのかを意識しなくても、要求仕様に従ったデータを利用することができる。

【００２０】

また、例えばデータのサンプリングタイミングも請求項４のようにして、結合処理プログラムによる処理で調整するように、結合処理部を構成するとよい。このようにすれば、アプリケーションプログラムの開発者は、アプリケーションプログラムの作成時に、プラットフォームプログラムによる処理におけるサンプリングタイミングがどのようになっているのかを意識することなく、要求仕様に従ったタイミングでデータを利用するように構成することができる。

【００２１】

そして、ＡＰインターフェースは、例えば、請求項５あるいは請求項６に示すようにしてインターフェースを行うようにすることができる。すなわち請求項５に示すようにして、変換したデータを、アプリケーションプログラムによる処理によって参照可能に構成するか、あるいは、請求項６に示すようにして、アプリケーションプログラムによる処理によってデータの要求があった場合に、変換したデータをアプリケーションプログラムによる処理に渡すように構成することができる。例えば、結合処理プログラムによる処理で予め定められたメモリアドレスのメモリ等の記憶手段にデータを記憶し、アプリケーションプログラムによる処理でその記憶手段を参照するようにしたり、あるいは、例えばいわゆるＡＰＩのように、アプリケーションプログラムによる処理から結合処理部による処理に移行して、結合処理部による処理で例えばスタック等へ要求されたデータまたはデータの格納先のアドレス（ポインタ等）を格納させて、アプリケーションプログラムによる処理へ戻り、スタック等からデータまたはデータの格納先のアドレ

スを取り出すように構成することができる。なお、ＰＦインターフェースもこうしたＡＰインターフェースと同様の方法によってインターフェースを行うことができる。

【００２２】

そして、結合処理プログラムは、ＰＦインターフェースを介して取得して変換したデータをそのままＡＰインターフェースを介して提供するように構成してもよいが、例えば請求項７に示すように、ＰＦインターフェースを介して取得したデータを記憶手段に記憶しておき、その記憶手段に記憶されたデータを変換したデータをＡＰインターフェースを介して提供するプログラムとしたり、ＰＦインターフェースを介して取得したデータを加工したデータを記憶手段に記憶しておき、その記憶手段に記憶されたデータをＡＰインターフェースを介して提供するプログラムとしたりするとよい。

【００２３】

例えば請求項８に示すように、プラットフォームプログラムには、ハードウェアデバイスからの入力をハードウェアデバイスからの割り込みによって行い、その入力結果に基づくデータをＰＦインターフェースを介して提供する処理のためのプログラムを備え、結合処理プログラムは、さらに、他の処理のディスパッチを禁止している間に、ＰＦインターフェースを介してデータを受け取り、記憶手段に記憶する処理をさせるための取得プログラムを備えるとよい。このようにすれば、取得部による「ＰＦインターフェースを介してデータを受け取り、記憶手段に記憶する処理」を行っている間は、他の処理のディスパッチが禁止されるので、データの一貫性を保つことが容易になる。また、アプリケーションプログラムによる処理では、例えば、この記憶手段に記憶されたデータをＡＰインターフェース介して取得して利用することができる。

【００２４】

以上、ハードウェアデバイスからデータの入力処理を行う場合の車両用制御プログラムの構成について説明したが、ハードウェアデバイスへ出力を行う場合にも、同様に構成するとよい。すなわち、入力処理の場合と同様に、車両用制御プログラムをプラットフォームプログラムと、結合処理部と、アプリケーションプ

ログラムとに分けで構成する。例えば、請求項９に示すようにプラットフォームプログラムは、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースであるＰＦインターフェースに従って出力のためのデータを前記結合処理プログラムによる処理から取得し、取得したデータに基づくデータを出力する処理をコンピュータに実行させるためのプログラムとし、結合処理プログラムは、開発対象の車両用制御装置の要求仕様を満たすインターフェースであるＡＰインターフェースに従ってアプリケーションプログラムによる処理から提供されたデータを、ＰＦインターフェースに適合するように変換して、プラットフォームプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムとし、アプリケーションプログラムは、出力対象のデータを生成し、ＡＰインターフェースに従って前記出力対象のデータを結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムとする。このようにすれば、アプリケーションプログラムの開発者は、ＰＦインターフェースの仕様を意識することなく、単にＡＰインターフェース仕様に従ってデータを出力するようにプログラムを作成するだけでよい。すなわち、要求仕様に従ったアプリケーションプログラムを作成するだけでよいため、アプリケーションプログラムの開発者の負担を軽減することができる。

【００２５】

ＡＰインターフェースの構成としては種々の構成を採りうるが、例えば、請求項１０に示すようにして、ＡＰインターフェースは、前記アプリケーションプログラムによる処理によって記憶されたデータを結合処理プログラムによる処理によって自ら取得する構成とするとよい。そして、例えば、取得したデータに基づくデータをプラットフォームプログラムによる処理に渡してハードウェアデバイスへの出力処理をさせるように構成するとよい。このようにすれば、アプリケーションでは、単に演算結果（例えば駆動量や駆動タイミング）のデータを求めて記憶手段に記憶しておくだけで、結合処理プログラムによる処理によって、そのデータが自動的に取得され、プラットフォームプログラムの仕様を満足する適切なタイミングやデータ型式で、プラットフォームプログラムによる処理へ仲介さ

れる。したがって、アプリケーションの開発者は、アプリケーションの開発をスムーズに行うことができる。

【００２６】

なお、結合処理プログラムは、入力あるいは出力のいずれかのみについて仲介を行うように構成してもよいが、入力と出力の双方を仲介するようにするとよい。例えば、請求項１１に示すように、請求項１～８のいずれかに記載の車両用プログラムと請求項９または１０に記載の車両用制御用プログラムとを備える車両用制御プログラムとするとよい。このようにすれば、アプリケーションプログラムの開発者は、入出力の双方について、プラットフォームプログラムによる処理によって提供されるＰＦインターフェースの仕様を意識することなく、単に要求仕様に従ってアプリケーションプログラムを作成することができる。

【００２７】

そして、請求項１２に示すように、請求項１～１１のいずれかに記載の車両用制御プログラムと、この車両用制御プログラムを実行するコンピュータとを備えた車両用制御装置を構成することができる。このような車両用制御装置は、アプリケーションの開発を効率的に行うことができるため、従来より、低コストでの開発が容易にできるようになる。

【００２８】

なお、請求項１に記載の車両用制御プログラムは、例えば請求項１３に示す生成方法によって、生成（製造）することができる。（請求項１３は、物（プログラム）を生産する方法の発明である。）なお、請求項１３に示す生成方法におけるプラットフォームプログラムと結合処理部とアプリケーションの生成の順序はどのような順序でも構わない。

【００２９】

例えば、新たな開発対象の車両用制御装置がある場合、プラットフォームプログラムについては、すでに開発済みの車両用制御装置のものをできるだけ再利用することとし、車両メーカからの要求仕様を満足しない部分のみを修正して、ＰＦインターフェース仕様を決定する。そして、車両メーカからの要求仕様に従って、ＡＰインターフェースの仕様であるＡＰインターフェース仕様を決定する。

このようにＰＦインターフェース仕様とＡＰインターフェース仕様を、新たな開発対象の車両用制御装置の開発の初期段階で決定すれば、プラットフォームプログラムと結合処理部とアプリケーションプログラムとを並行して開発することも可能となる。（なお、請求項２～１２に記載の車両用制御プログラムについても、請求項１に対する請求項１３の記載のようにして、生成方法の請求項とすることも可能である。）

【００３０】

【発明の実施の形態】

以下、本発明が適用された実施例について図面を用いて説明する。なお、本発明の実施の形態は、下記の実施例に何ら限定されることなく、本発明の技術的範囲に属する限り種々の形態を採りうることは言うまでもない。

【００３１】

本実施例の車両用制御装置は、エンジンの制御を行うためのエンジンＥＣＵであり、図１に本実施例の説明に必要な部分についての構成を図示している。本実施例のエンジンＥＣＵ１０と従来のエンジンＥＣＵとの差異は、ＲＯＭ１２に記憶されたプログラムの構成にあり、その他の構成は従来のエンジンＥＣＵと同様である。

【００３２】

エンジンＥＣＵ１０は、図１に示すように、ＣＰＵ１１，ＲＯＭ１２，ＲＡＭ１３，Ｉ／Ｏ１４、Ａ／Ｄ変換器１５、タイマ１６等とこれらを接続するバスライン等から構成されている。そして、Ｉ／Ｏ１４やＡ／Ｄ変換器１５には、エンジン回転センサ２１、Ｏ２センサヒータ２２、水温センサ２３、…、といった各種のエンジン制御用装置２０が駆動回路等を介して接続されている。そして、ＲＯＭ１２に記憶されたプログラムをＣＰＵ１１が実行することによって、こうしたエンジン制御用装置２０からの信号の入力や、エンジン制御用装置２０に対する信号の出力を行う。例えば、ＣＰＵ１１による処理によって、エンジン回転センサ２１からのクランク軸の所定回転角毎（例えば１８０度毎）に発生するパルスをＩ／Ｏ１４から入力したり、Ｉ／Ｏ１４に対してＯ２センサを所定の動作温度まで加熱するためのＯ２センサヒータ２２をＰＷＭ制御するＯ２センサヒータ

制御信号を出力したりする。また、Ａ／Ｄ変換器１５を介して、水温センサ２３から水温に対応する電圧値を入力し、入力した電圧値をデジタルデータとしてＲＡＭ１３等へ取り込む。

【００３３】

こうした制御をＣＰＵ１１に実行させるためにＲＯＭ１２に記憶するプログラム１は、図２（ａ）に示すように、プラットフォームプログラム２（図中ＰＦで表記している）と、特許請求の範囲における結合処理プログラムに相当する結合処理部３と、アプリケーションプログラム４（図中ＡＰで表記している）とをリンクして構成している。

【００３４】

プラットフォームプログラム２は、Ｉ／Ｏ１４やＡ／Ｄ変換器１５等のハードウェアデバイスを制御する処理（ハードウェアに依存する処理）をＣＰＵ１１が実行するためのデバイスドライバを含むプログラムであり、一方アプリケーション４は、Ｉ／Ｏ１４やＡ／Ｄ変換器１５といったハードウェアデバイスを直接制御する処理を行わず、車両メーカからの要求仕様を実現するための処理（車両用の制御に依存する処理（判定処理、演算処理など））をＣＰＵ１１が実行するためのプログラムである。そして、結合処理部３は、アプリケーションプログラム４によるＣＰＵ１１の処理とプラットフォームプログラム２におけるＣＰＵ１１の処理との仲介を行うための処理をＣＰＵ１１が実行するためのプログラムである。

【００３５】

プラットフォームプログラム２と結合処理部３とアプリケーションプログラム４は別々に開発が行われており、プラットフォームプログラム２は、すでに開発済みの従来のエンジンＥＣＵに用いられていたものを利用する。一方、結合処理部３とアプリケーションプログラム４は新規に開発する。

【００３６】

プラットフォームプログラム２は、図２（ｂ）に示すように、結合処理部３による処理からのＩ／Ｏ１４やＡ／Ｄ変換器１５の駆動要求を受け付けてＩ／Ｏ１４やＡ／Ｄ変換器１５を駆動し、また、Ｉ／Ｏ１４やＡ／Ｄ変換器１５から取得

した情報を結合処理部３による処理から参照可能とする処理を行うことによって、結合処理部３の処理に対するインターフェースを提供する（ＰＦインターフェースと称する）。このＰＦインターフェースは、すでに開発済みの従来の車種のエンジンＥＣＵの要求仕様を満足するように標準化して構成されている。

【００３７】

また、結合処理部３は、ＰＦインターフェースを介して取得した情報を開発対象のアプリケーションプログラム４の要求仕様を満足する情報に変換してアプリケーションプログラム４の処理から参照可能とする処理と、アプリケーションプログラム４による処理によって生成された要求仕様に従った出力対象の情報を参照してＰＦインターフェースの要求する情報に変換してＰＦインターフェースを介してＩ／Ｏ１４へ出力させる処理をＣＰＵ１１に行わせるプログラムである。

【００３８】

こうした結合処理部３を設けることによって、アプリケーションプログラム４の開発者は、ＰＦインターフェースの仕様を意識することなく、要求仕様のみを考慮してアプリケーションプログラムを開発することができる。この要求仕様には、入出力するデータのタイミングやデータ型式等のＡＰインターフェースに関する仕様と、ＡＰインターフェースを介して取得したデータを利用した車両用の制御仕様とが含まれる。制御仕様については、アプリケーションプログラム２の開発中であってもたびたび車両メーカ側からの変更の要求があるのに対し、ＡＰインターフェースに関する仕様については、開発の初期段階でほぼフィックスされている。したがって開発の初期段階でＡＰインターフェースの仕様は決定することができ、また、ＰＦインターフェースの仕様はすでに標準化されているものを流用するため、開発の初期段階から結合処理部３の開発を進めることができる。一方、アプリケーションプログラム２については、開発の初期段階から要求仕様のみを意識して開発を進めることができ、開発中に制御仕様の変更があったとしても、ＡＰインターフェースは変更する必要がなく、制御仕様に従って制御部分の変更を行うだけで済む。

【００３９】

こうした結合処理部３を備えたプログラム１の詳細な構成の例を図に示して説

明する。

例えば、図３（ａ）に示すように、ＰＦインターフェースによってunsigned short型でＬＳＢの示す値が５／６５５３６［Ｖ］のデータ型式でデータが提供されるよう標準化されている一方、要求仕様のデータ型式がunsigned char型でＬＳＢの示す値が５／２５６［Ｖ］の場合について説明する。

【００４０】

この場合、図３（ｂ）に示すように、結合処理部３は、プラットフォームプログラム２による処理から受けとったデータのデータ型及びＬＳＢの対応する値についての変換処理をＣＰＵ１１に実行させるように構成する。

すなわち、結合処理部３は、ＰＦインターフェースを介してunsigned short型でＬＳＢの示す値が５／６５５３６［Ｖ］のデータを取得し、取得したデータをunsigned char型でＬＳＢの示す値が５／２５６［Ｖ］のデータに変換（加工）する変換処理を行い、変換したデータをアプリケーションプログラム４による処理から参照可能にする処理をＣＰＵ１１が実行するための命令コードを備える。例えば結合処理部３による処理では、変換したデータを予め定めたアドレスのＲＡＭ１３に格納しておき、アプリケーションプログラム４による処理では、このアドレスのＲＡＭ１３の値を取得するようにして、ＡＰインターフェースを構成し、アプリケーションプログラム４による処理による要求仕様に沿ったデータの参照を行うことができる。

【００４１】

従来の車両用制御プログラムの構成では、図４に示すように、プラットフォームプログラム２による処理によって提供されるunsigned short型でＬＳＢの示す値が５／６５５３６［Ｖ］のデータ型式のデータをアプリケーションプログラムによる処理で取得して、アプリケーションプログラムによる処理で、要求仕様を満たすデータ型式であるunsigned char型のＬＳＢの示す値が５／２５６［Ｖ］のデータに変換する変換処理を行う必要があった。したがって、このような変換処理をアプリケーションプログラム内で行う必要があり、アプリケーションプログラムの開発者は、この点を常に意識しながらアプリケーションプログラムを開発する必要があるため、アプリケーションプログラムの開発者の負担が大きかっ

た。しかし、上述した結合処理部３を備えることにより、アプリケーションプログラム４の開発者は、ＰＦインターフェースのデータ型式を意識することなく、単に要求仕様のみを考慮してアプリケーションプログラム４の開発を行うことができる。

【００４２】

次に、結合処理部３による処理で、データの取得タイミングを調整する例を示す。

図５（ａ）に示すように、プラットフォームプログラム２による処理で提供されるＰＦインターフェースは、結合処理部３による処理から参照可能なデータを１ｍｓ毎に更新する構成で標準化されており、一方、要求仕様としては、８ｍｓ毎に利用する仕様である。但し、この８ｍｓ毎に利用するデータは、１６ｍｓ毎に取り込んだデータとする。

【００４３】

この場合、図５（ｂ）に示すように、プラットフォームプログラム２は、１ｍｓ毎に水温センサ２３の電圧値をＡ／Ｄ変換器１５を介してプラットフォームプログラム２の管理下のＲＡＭ１３に取り込む処理をＣＰＵ１１に実行させるための命令コードを備える。またプラットフォームプログラム２は、結合処理部３による処理からの要求に応じてプラットフォームプログラム２による管理下のＲＡＭ１３に取り込んだ電圧値を結合処理部３による処理へ渡す処理をＣＰＵ１１に実行させるための命令コードを備える。

【００４４】

そして、結合処理部３は、１６ｍｓ毎にプラットフォームプログラム２による処理に対して電圧値を要求して、プラットフォームプログラム２による処理における管理下のメモリに取り込まれた電圧値を受け取り、結合処理部３の処理による管理下のメモリに記憶する処理をＣＰＵ１１に実行させるための命令コードを備える。また、アプリケーションプログラム４による処理からの要求に応じて、この結合処理部３による処理の管理下のメモリに記憶された電圧値をアプリケーションプログラム４による処理へ渡す処理を行うための命令コードを備えることで、ＡＰインターフェースを提供する。

【００４５】

そして、アプリケーションプログラム４は、８ｍｓ毎に結合処理部３に対して電圧値を要求し、結合処理部３による処理から電圧値を受け取って、利用する処理を行う命令コードを備える。

なお、アプリケーションプログラム４による処理からこの結合処理部３による処理の管理下のメモリを直接参照して取得するＡＰインターフェースとしてもよい。

【００４６】

また、プラットフォームプログラム２には、結合処理部３を１６ｍｓ以内の間隔で実行し、アプリケーション４を８ｍｓ以内の間隔で実行するための命令コードを備える。

こうした車両用制御プログラム１をＣＰＵ１１が実行すると、図５（ｂ）に示すように、結合処理部３による処理で、１６ｍｓ毎に電圧値をＰＦインターフェースを介して取得して記憶しておき、アプリケーションプログラム４の処理によって８ｍｓ毎に、この記憶している電圧値を利用する。

【００４７】

このように、結合処理部３を設けることで、アプリケーションプログラム４は、要求仕様を満足する１６ｍｓ毎にサンプリングされた電圧値を、８ｍｓ毎に利用することができる。すなわち、アプリケーションの開発者は、図６に示す従来例のようにアプリケーションプログラムによる処理内でサンプリングタイミング変換しなければならないことを意識する必要はなく、アプリケーションプログラム４を開発することができる。

【００４８】

次に、割り込み処理に適用した例を説明する。図７に示すように、エンジン回転センサ２１からの信号パルスの立ち下がりで、ＣＰＵ１１にＩ／Ｏ１４から割り込みがかかり、ＣＰＵ１１はプラットフォームプログラム２に備えた割り込み処理ルーチンを実行する。このプラットフォームプログラム２の割り込み処理ルーチンは、この割り込み処理ルーチンの前回の実行時からの経過時間をタイマ１６の値を取得して算出し、これに基づいてエンジン回転数を求めて、プラットフ

ォームプログラム2の管理下のＲＡＭ１３のメモリ領域へ記憶する処理を行うためのＣＰＵ１１の命令コードを備える。また、プラットフォームプログラム２は、結合処理部３による処理からの要求に応じてこのプラットフォームプログラム２の管理下のＲＡＭ１３のメモリ領域へ記憶された回転数を結合処理部３による処理に渡す処理を行うためのＣＰＵ１１の命令コードを備える。そして、プラットフォームプログラム２は、結合処理部３のタスク（特許請求の範囲における取得プログラムに相当する）を起動する命令コードを備え、この命令コードをＣＰＵ１１が実行することで、結合処理部３のタスクが起動される。

【００４９】

このタスクは、プラットフォームプログラム２の処理に回転数を要求するための命令コードと、プラットフォームプログラム２による処理から受け取った回転数を結合処理部３の管理下のＲＡＭ１３のメモリ領域へ記憶する処理を行うための命令コードと、アプリケーションプログラム４による処理から回転数の要求があった場合に、この結合処理部３の管理下のＲＡＭ１３のメモリ領域へ記憶された回転数をアプリケーションプログラム４による処理へ渡す処理を行うための命令コードとを備える。

【００５０】

そして、エンジン回転数を利用して要求仕様を満たす機能を実現させるためのプログラムであるアプリケーションプログラム４の回転数利用処理部は、エンジン回転数を参照する各部で、結合処理部３による処理に対して回転数を要求するための命令コードを備える。

【００５１】

そして、割り込み処理部の実行レベルは他のプログラム部分よりレベルを高く設定し、結合処理部３の実行レベルはアプリケーションプログラム４の実行レベルと同一かそれ以下に設定する命令コードをプラットフォームプログラム２に備える。そして、ＣＰＵ１１は、事前にこの実行レベルの設定をするプログラムの命令コードを実行しておく。また、プラットフォームプログラム２には、実行レベルに応じて実行すべきタスクを決定する処理を行う命令コードを備える。

【００５２】

このような構成によれば、図７に示すように、エンジン回転センサ信号のパルスの立ち下がりで、ＣＰＵ１１がプラットフォームプログラム２の割り込み処理ルーチンを実行し、エンジン回転数を求めて、プラットフォームプログラム２の管理下のＲＡＭ１３のメモリ領域へ記憶する。そして、プラットフォームプログラム２による処理によって、アプリケーションプログラム４が実行中でない場合には、図７に示す割り込み処理Ｐ１の後のように、結合処理部３のタスク処理部Ｃ１が実行されてプラットフォームプログラムによる処理に対して回転数を要求し、プラットフォームプログラムによる処理では結合処理部３による処理に対して回転数を渡す。結合処理部３のタスク処理部Ｃ１の処理により回転数を受け取り、結合処理部３の管理下のＲＡＭ１３のメモリ領域へ記憶する。そしてプラットフォームプログラム２は、アプリケーションＡ１を実行する。

【００５３】

アプリケーションＡ１において、回転数を参照する場合には、結合処理部３による処理に回転数を要求する。結合処理部３による処理では、結合処理部３の管理下のＲＡＭ１３のメモリ領域へ記憶された回転数を渡す。そして、アプリケーションプログラム４の処理では、この回転数を受け取って自動車用の制御処理に利用する。

【００５４】

また、図７に示すように、アプリケーションＡ２の処理中に、エンジン回転センサ信号による割り込みが発生し、プラットフォームプログラム２の割り込み処理Ｐ３が実行される場合、割り込み処理Ｐ３の完了後、プラットフォームプログラム２によって結合処理部Ｃ３が起動されるのであるが、結合処理部Ｃ３の実行レベルは、アプリケーションＡ２の実行レベルと同一かそれ以下に設定されているため、プラットフォームプログラム２による処理では、結合処理部Ｃ３の実行を後回しにして、先に、アプリケーションＡ２の割り込みによる中断時点からの命令を実行する。そのため、アプリケーションＡ２では、途中で割り込み処理Ｐ３を行っているが、回転数を参照する場合には、結合処理部Ｃ２の管理下のＲＡＭ１３のメモリ領域へ記憶された回転数を取得して処理に利用することとなる。

【００５５】

このように、アプリケーションプログラム２による処理で結合処理部３による処理を介して回転数を取得するようにプログラムを構成することで、アプリケーションプログラム４の開発者は、プラットフォームプログラム２の仕様を意識することなく、単に結合処理部３とのやりとりを行うＡＰインターフェースの仕様を知るだけでアプリケーションプログラム４を開発することができる。

【００５６】

すなわち、従来は、上述した結合処理部３が存在しないため、図８（ａ）のようにアプリケーションＡ１の処理中で、回転数の参照を行うと、プラットフォームの割り込み処理Ｐ１でメモリに記憶された回転数がプラットフォームから提供されていた。この場合、アプリケーションＡ１中で複数回のプラットフォームの回転数を参照したとしても、アプリケーションＡ１中では、得られる回転数は全て同じ値となり、問題は生じない。しかし、割り込み処理Ｐ２の後にアプリケーション処理Ａ２が実行され、このアプリケーションＡ２中に、エンジン回転センサ信号による割り込みが発生した場合、アプリケーションＡ２の処理は中断され、プラットフォームプログラム２による割り込み処理Ｐ３が実行される。そして、割り込み処理Ｐ３の完了後にアプリケーション処理Ａ２が再開される。この結果、割り込み処理Ｐ３中で回転数は新たな値に更新されるので、中断前のアプリケーション処理Ａ２の部分では、プラットフォームから取得した割り込み処理Ｐ２による回転数を用いた処理がなされ、割り込み処理Ｐ３から復帰した後のアプリケーション処理Ａ２の部分では、プラットフォームから取得した割り込み処理Ｐ３による回転数を用いた処理がなされることとなる。このように、１単位のアプリケーション処理内で、ハードウェアデバイスからのデータに基づくデータに関して一貫性が保持できない。そのため、従来は、アプリケーションの開発者は、こうしたプラットフォームの制約条件を理解し、例えば図８（ｂ）に示すように、割り込みの影響を受けないように、アプリケーション処理の先頭でディスパッチ禁止処理を行い、プラットフォームのＡＰＩを呼び出して回転数を取り込んで、自己の管理下のメモリに記憶する（グローバルコピーすると称する）処理を行い、そして、それ以降のアプリケーション処理では、このグローバルコピーされた回転数を用いて処理を行うようにする必要があった。このように、アプリケ

ーションプログラムの開発者は、プラットフォームの仕様（制約条件）を熟知し、仕様間のギャップを埋めて、アプリケーションプログラムを作成しなければならない。もしも、こうしたプラットフォームの仕様をアプリケーション作成者がよく理解せずに作成すると、例えば図８（ａ）のように、ときどき動作がおかしくなるといった症状などが生じ、作成したアプリケーションプログラムが、アプリケーション作成者の意図通りに動かない。このような場合、アプリケーションの開発者はプラットフォームの仕様を理解していないため、動作異常の原因を特定するのは困難になってしまう。かといって、複数のアプリケーション（例えばタスク）を動作させる場合、すべてのアプリケーションについてグローバルコピーを行うとメモリ領域の消費や処理時間の消費などオーバーヘッドが大きくなる。そのため、どのアプリケーションが、処理内でのデータの一貫性（同時性）を保てないのかを、プラットフォームの仕様とアプリケーションの仕様とを照らし合わせて判断する必要があり、アプリケーションの開発者の負担が大きく、開発効率を高める上での障害となっていた。

【００５７】

しかし、図７に示して説明した本実施例の構成にすれば、図８（ｂ）に示すようなグローバルコピーをアプリケーションプログラム中で行う必要もなくなり、どのアプリケーションプログラムについてグロバールコピーをする必要があるのかなどを検討する必要もなくなる。したがって、アプリケーションの生産性を向上させることができる。

【００５８】

以上、アプリケーションプログラムによる処理でデータを入力する場合について説明したが、アプリケーションプログラムによる処理で、データを出力する場合にも、従来のように、直接プラットフォームプログラムの提供するインターフェースにアクセスするのではなく、結合処理部３を介して処理を行うとよい。

【００５９】

例えば、図９（ａ）に示すように、アプリケーションプログラム４は、条件判定処理と、演算処理をＣＰＵ１１が実行するための命令コードを備える。そして、結合処理部３は、プラットフォームプログラム２による処理によって、例えば

所定周期毎に実行される。

【００６０】

そして、結合処理部３には、アプリケーションプログラム２による演算処理によって得られたデータを取得する情報取得処理と、情報取得処理によって取得したデータをプラットフォームプログラム２による負荷駆動処理で用いるデータ型式に変換する情報変換処理と、プラットフォームプログラム２による負荷駆動処理を実行させるための負荷駆動要求処理とをＣＰＵ１１が実行するための命令コードを備える。また、プラットフォームプログラム２は、負荷駆動要求があった場合に、負荷駆動処理を行う。

【００６１】

例えば、図９（ｂ）に示すように、エンジンを始動して所定時間経過後、エンジン始動時の水温に応じてＯ２センサヒータ２２の制御を行い、要求仕様として周期２５６ｍｓでＤｕｔｙ制御するものであるとする。

この場合、プラットフォームプログラムでは、図９（ｂ）に示すように、Ｏ２センサヒータ２２を駆動するプラットフォームのＡＰＩをＯ２Ｈ＿ＤｕｔｙＯｕｔ（周期，Ｄｕｔｙ）とし、周期のデータ型は、unsigned shortでＬＳＢの対応する値は１ｍｓとし、Ｄｕｔｙのデータ型はsigned longでＬＳＢの対応する値は１／６５５３６とする。Ｄｕｔｙのデータ範囲は０～１として、それ以外は動作を保証しない。

【００６２】

一方、要求仕様は、アプリケーションプログラム４の演算処理によって出力するＤｕｔｙがfloat型であり、範囲が０～１００％で、システムの要求する周期２５６ｍｓでＤｕｔｙ制御であるとする。

このとき、従来は、アプリケーションプログラムでは、図１１及び図１２に示すように、アプリケーション内の演算処理で出力されたfloat型のＤｕｔｙを、プラットフォームが指定する型およびＬＳＢの対応する値に変換処理して、プラットフォームのＡＰＩを呼び出すように構成する必要があった。

【００６３】

具体的には、アプリケーションプログラムとプラットフォームプログラムとは

それぞれ図１２に示すように、処理を行っていた。すなわち、アプリケーションプログラムは、図１２に示すように条件判定処理、演算処理、負荷駆動要求処理を行い、プラットフォームは、負荷駆動処理を行う。

【００６４】

アプリケーションプログラムでは、Ｓ１１０で、エンジン始動後所定時間経過したか否かを判定し、所定時間経過した場合には（Ｓ１１０：成立）、Ｓ１２０へ移行して、所定の処理（始動時水温補間テーブルによる補間）によってＤｕｔｙ＿ＡＰを求める。一方、エンジン始動後所定時間経過していない場合には（Ｓ１１０：不成立）、ヒータ制御は不要であるので、Ｓ１３０へ移行して、Ｄｕｔｙ＿ＡＰを０％にする。

【００６５】

そしてＳ１４０では、Ｄｕｔｙ＿ＡＰを、プラットフォームのＡＰＩ（プラットフォーム命令）の引数として用いられる型とＬＳＢの示す値に変換してＤｕｔｙ＿ＰＦとして求める。すなわち、変換後のＬＳＢの示す値が１／６５５３６となるようにfloat型からsigned long型への変換を行う。続くＳ１５０では、Ｄｕｔｙ＿ＰＦをプラットフォームの仕様に従って０～１でガードする。そして、Ｓ１６０で、周期を２５６（ｍｓ）、ＤｕｔｙをＤｕｔｙ＿ＰＦとしてプラットフォーム命令Ｏ２Ｈ＿Ｄｕｔｙを呼び出す。このようにアプリケーションの開発者は、プラットフォームの制約条件を意識して、アプリケーション内に変換処理を設ける必要がある。

【００６６】

プラットフォーム側では、この呼び出しに応じて、呼び出し時にセットされた値（引数）に基づき、実際にハードウェアを制御し、ヒーター制御を行う（Ｓ２００）。

このように、プラットフォームプログラムの提供する機能をアプリケーションプログラムから直接利用するため、プラットフォームプログラムの動作上の仕様が、アプリケーションプログラム作成時の制約条件となってしまう。アプリケーションプログラムの開発者は、常にこうした制約条件に注意を払って、入出力のタイミングやデータ形式等に気を配ってこうした制約条件をなくすためのルーチ

ン（例えばＣＰＵ１１の命令コード）をアプリケーションに組み込む必要があった。

【００６７】

しかも、アプリケーションプログラムは、機能の追加、変更等の頻度が高く、こうした制約条件に注意を払いながら、これらの機能の追加や変更を行うのは困難であった。

そこで、本実施例では、図１２に示した従来のプログラム構成を図１０に示すように変更する。すなわち、上述したように従来は、条件判定処理と、演算処理と、負荷駆動要求処理とをアプリケーションプログラムで行うようにしていたのであるが、本実施例では、図１０に示すように結合処理部３に、従来、アプリケーションプログラムによって行っていた図１２のＳ１４０～１６０に相当する処理を、図１０のＳ４２０～４４０に示すように結合処理部３に設ける。そして結合処理部３には、アプリケーションプログラム４のＳ３１０～３３０によって求められたＤｕｔｙＡＰを読み込む処理（Ｓ４１０）を設ける。このようにして図１０に示すように、情報加工処理と負荷駆動要求処理を結合処理部３による処理で行う。

【００６８】

その結果、アプリケーションプログラム４は、図１０のＳ３１０～３３０のように、条件判定処理と演算処理とをＣＰＵ１１が実行するように構成するだけで済む。

このように、出力処理に関しても、アプリケーションの開発者は、プラットフォームプログラムのインターフェース仕様を気にすることなく、アプリケーションの開発を行うことができる。

【００６９】

なお、本実施例では、エンジンＥＣＵの例で説明したが、種々の車両用のＥＣＵのプログラム開発に適用できる。

【図面の簡単な説明】

【図１】実施例の車両用制御装置の構成を示すブロック図である。

【図２】実施例の車両用制御プログラムの構成を示す説明図である。

【図３】実施例の車両用制御プログラムによるデータ型式の変換処理の例を示す説明図である。

【図４】従来の車両用制御プログラムによるデータ型式の変換処理の例を示す説明図である。

【図５】実施例の車両用制御プログラムによるサンプリングタイミングの例を示す説明図である。

【図６】従来の車両用制御プログラムによるサンプリングタイミングの例を示す説明図である。

【図７】実施例の車両用制御プログラムによる割り込み処理の実行例を示す説明図である。

【図８】従来の車両用制御プログラムによる割り込み処理の実行例を示す説明図である。

【図９】実施例の車両用制御プログラムによるデータの出力処理の例を示す説明図である。

【図１０】実施例の車両用制御プログラムによるデータの出力処理を示すフローチャートである。

【図１１】従来の車両用制御プログラムによるデータの出力処理の例を示す説明図である。

【図１２】従来の車両用制御プログラムによるデータの出力処理を示すフローチャートである。

【符号の説明】

１…プログラム

２…プラットフォーム

３…結合処理部

４…アプリケーション

１０…エンジンＥＣＵ

１１…ＣＰＵ

１２…ＲＯＭ

１３…ＲＡＭ

１４…Ｉ／Ｏ

１５…Ａ／Ｄ変換器

１６…タイマ

２０…エンジン制御用装置

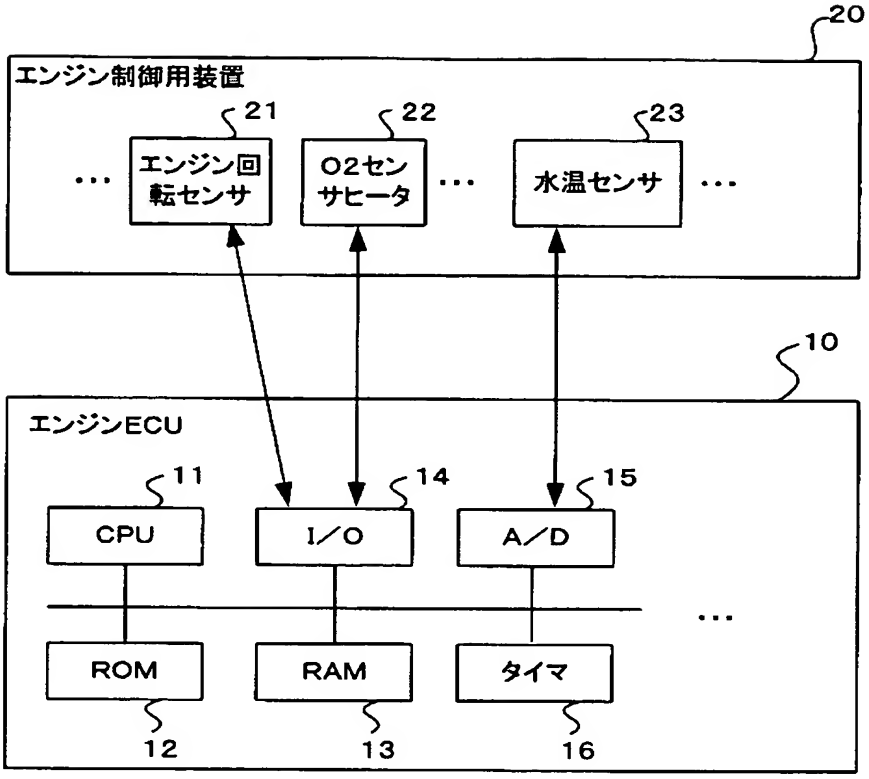
２１…エンジン回転センサ

２２…Ｏ２センサヒータ

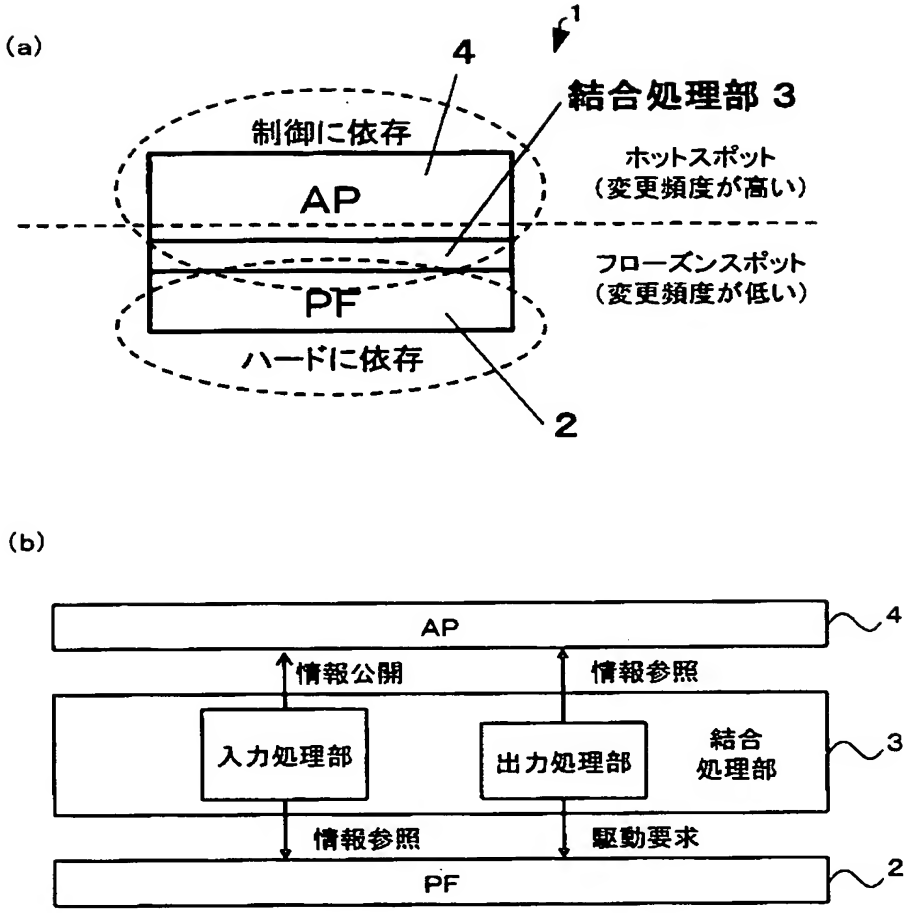
２３…水温センサ

【書類名】　図面

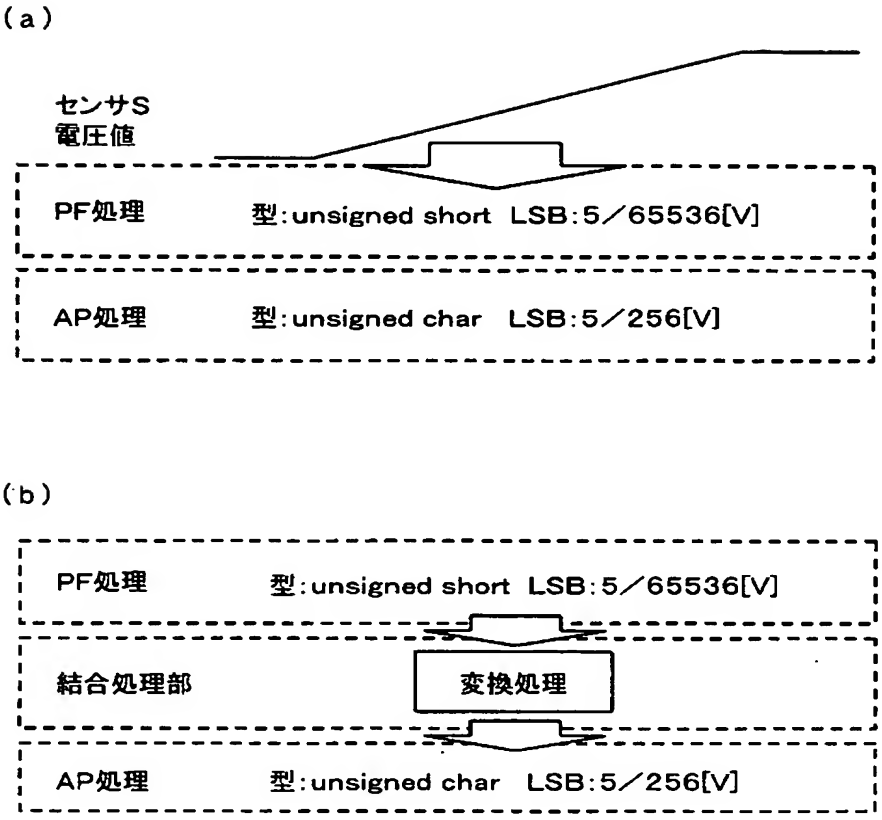
【図１】


20
エンジン制御用装置
21
22
23
エンジン回転センサ
O2センサヒータ
水温センサ
10
エンジンECU
11
14
15
CPU
I/O
A/D
ROM
RAM
タイマ
12
13
16

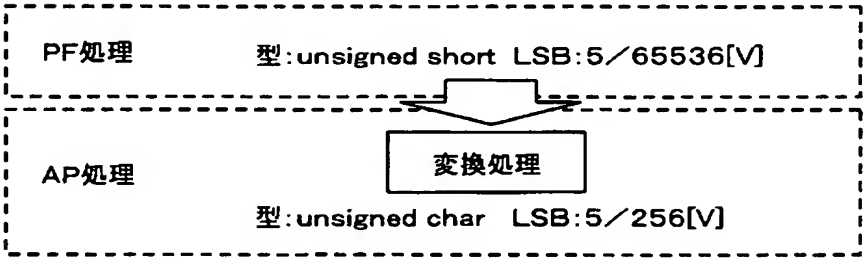
【図２】


(a)
1
4
結合処理部 3
制御に依存
AP
ホットスポット
(変更頻度が高い)
フローズンスポット
(変更頻度が低い)
PF
ハードに依存
2
(b)
AP
4
情報公開
情報参照
入力処理部
出力処理部
結合
処理部
3
情報参照
駆動要求
PF
2

【図３】

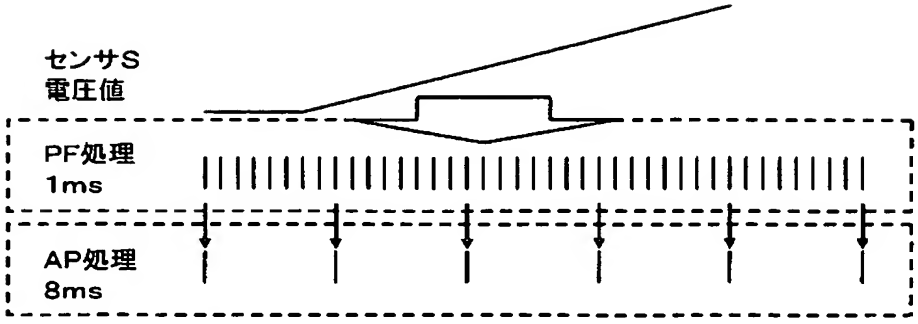

(a)
センサS
電圧値
PF処理
型:unsigned short LSB:5／65536[V]
AP処理
型:unsigned char LSB:5／256[V]
(b)
PF処理
型:unsigned short LSB:5／65536[V]
結合処理部
変換処理
AP処理
型:unsigned char LSB:5／256[V]

【図４】

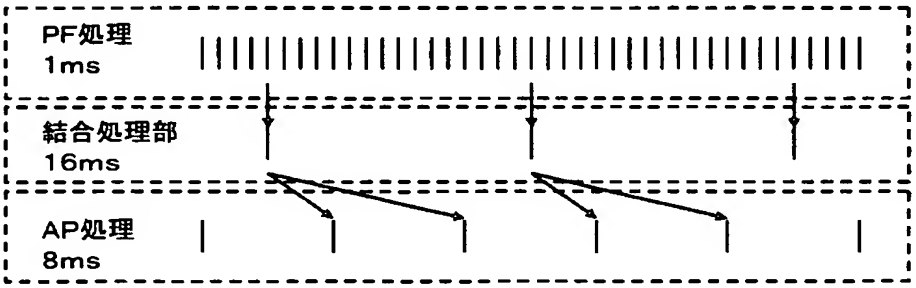

PF処理
型:unsigned short LSB:5／65536[V]
AP処理
変換処理
型:unsigned char LSB:5／256[V]

【図５】

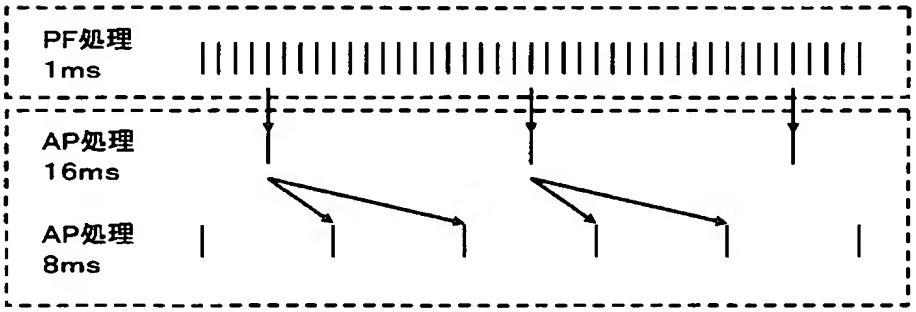
（a）


センサS
電圧値
PF処理
1ms
AP処理
8ms

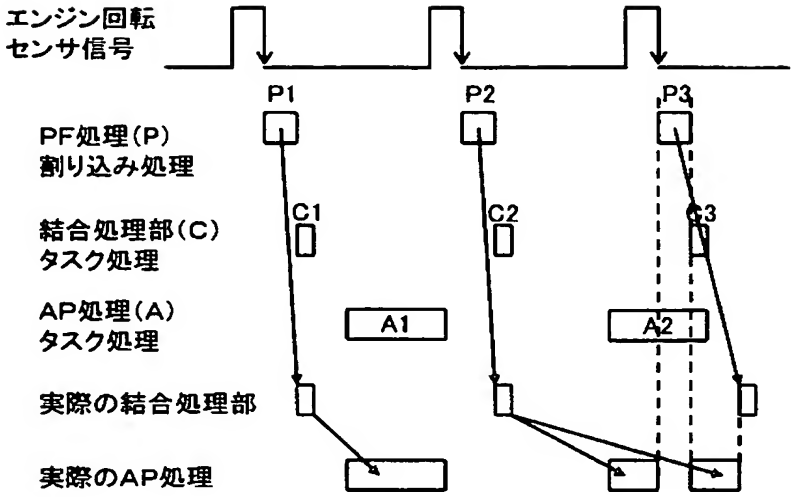
（b）


PF処理
1ms
結合処理部
16ms
AP処理
8ms

【図６】

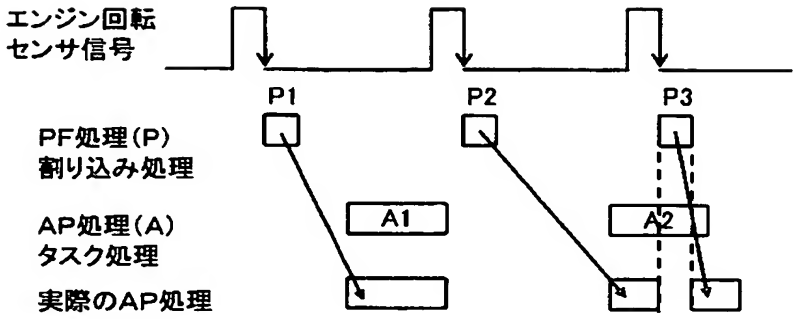

PF処理
1ms
AP処理
16ms
AP処理
8ms

【図７】

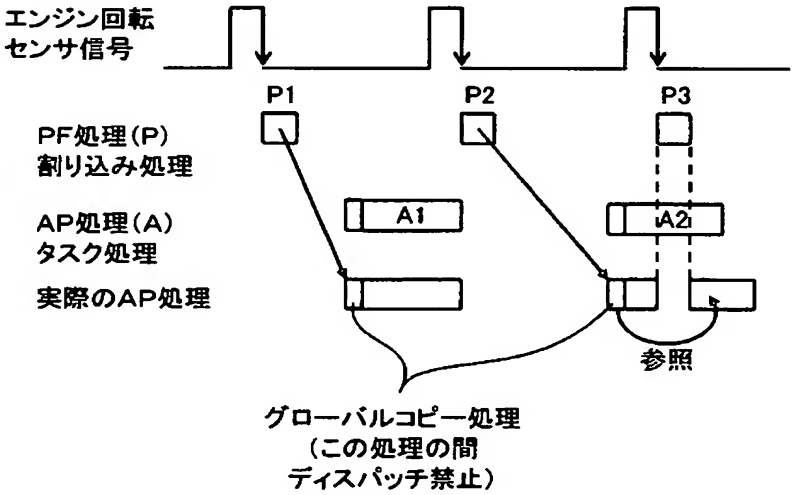

エンジン回転
センサ信号
P1
P2
P3
PF処理(P)
割り込み処理
結合処理部(C)
タスク処理
C1
C2
C3
AP処理(A)
タスク処理
A1
A2
実際の結合処理部
実際のAP処理

【図８】

(a)

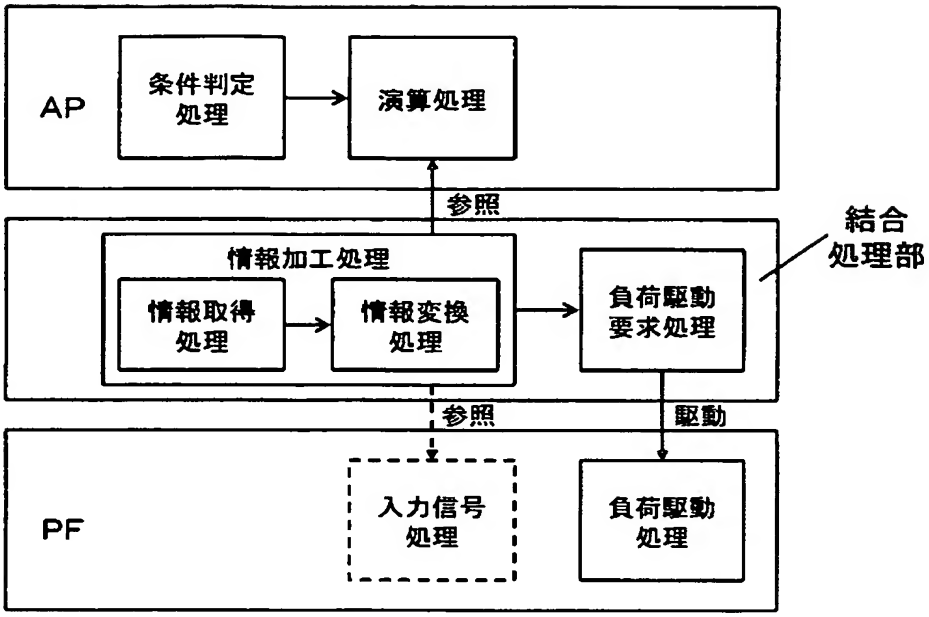

エンジン回転
センサ信号
P1
P2
P3
PF処理(P)
割り込み処理
AP処理(A)
タスク処理
実際のAP処理
A1
A2

(b)

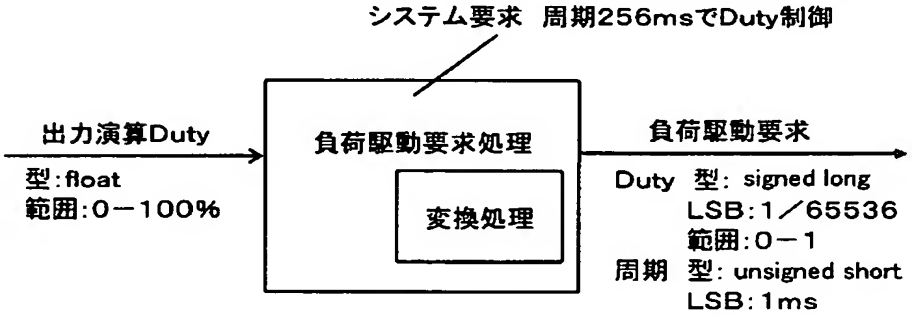

エンジン回転
センサ信号
P1
P2
P3
PF処理(P)
割り込み処理
AP処理(A)
タスク処理
実際のAP処理
A1
A2
参照
グローバルコピー処理
(この処理の間
ディスパッチ禁止)

【図９】

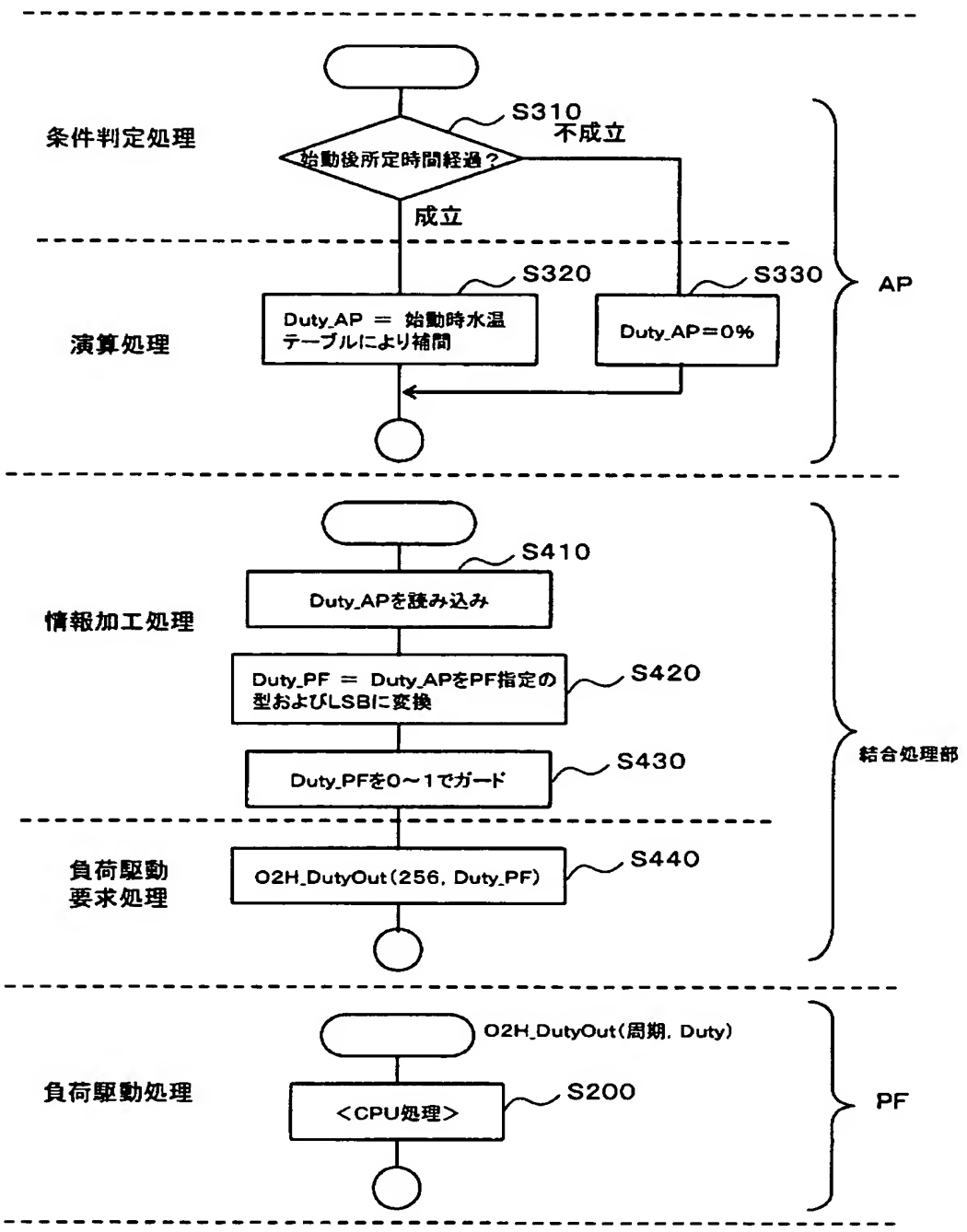
（ａ）


AP
条件判定処理
演算処理
参照
結合処理部
情報加工処理
情報取得処理
情報変換処理
負荷駆動要求処理
参照
駆動
PF
入力信号処理
負荷駆動処理

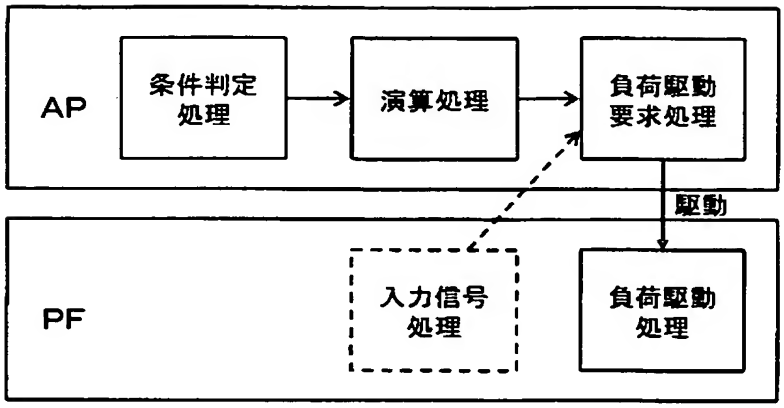
（ｂ）


システム要求　周期256msでDuty制御
出力演算Duty
型:float
範囲:0－100%
負荷駆動要求処理
変換処理
負荷駆動要求
Duty　型: signed long
LSB:1／65536
範囲:0－1
周期　型: unsigned short
LSB:1ms

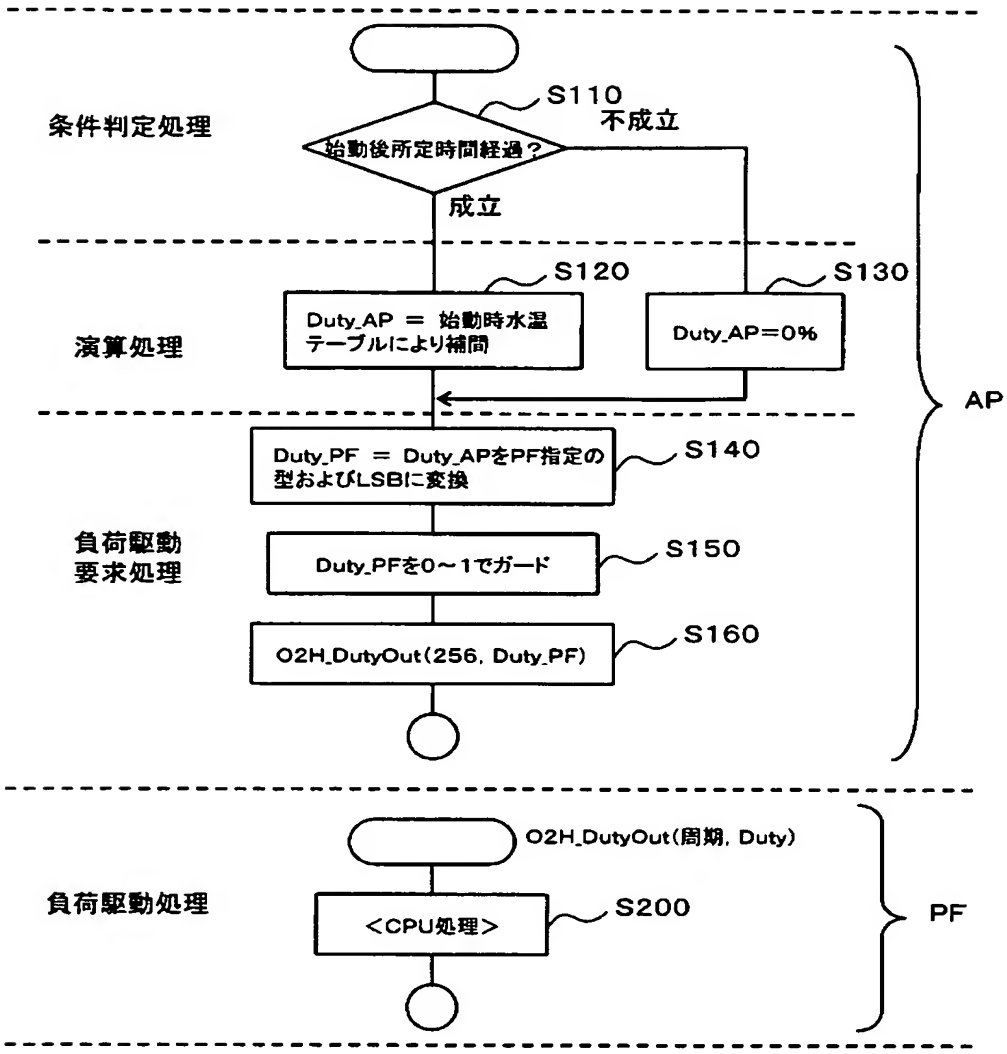
【図１０】


条件判定処理
S310
始動後所定時間経過？
不成立
成立
演算処理
S320
Duty_AP ＝ 始動時水温テーブルにより補間
S330
Duty_AP=0%
AP
情報加工処理
S410
Duty_APを読み込み
Duty_PF ＝ Duty_APをPF指定の型およびLSBに変換
S420
Duty_PFを0～1でガード
S430
負荷駆動要求処理
O2H_DutyOut(256, Duty_PF)
S440
結合処理部
負荷駆動処理
O2H_DutyOut(周期, Duty)
<CPU処理>
S200
PF

【図１１】



【図１２】


条件判定処理
S110
始動後所定時間経過？
不成立
成立
演算処理
S120
Duty_AP ＝ 始動時水温テーブルにより補間
S130
Duty_AP=0%
負荷駆動要求処理
Duty_PF ＝ Duty_APをPF指定の型およびLSBに変換
S140
Duty_PFを0～1でガード
S150
O2H_DutyOut(256, Duty_PF)
S160
AP
負荷駆動処理
O2H_DutyOut(周期, Duty)
<CPU処理>
S200
PF

【書類名】要約書

【要約】

【課題】アプリケーションプログラムの開発者の負担を軽減することのできる車両用制御プログラム等を提供する。

【解決手段】ハードウェアデバイスからの入力を行い、入力結果に基づくデータを他の処理に渡すための処理をコンピュータに実行させるためのデバイスドライバを含むプログラムであるプラットフォームプログラム２と、プラットフォームプログラム２による処理によって得られたデータをアプリケーションプログラム４による処理に適合するように仲介する処理をコンピュータに実行させるためのプログラムである結合処理部３と、プラットフォームプログラム２による処理によって得られたデータに基づく処理をコンピュータに実行させるためのプログラムであるアプリケーションプログラム４とを備える。

【選択図】図２

特願２００２－３６２４８７

出　願　人　履　歴　情　報

識別番号　　[０００００４２６０]

１．変更年月日　　１９９６年１０月　８日
[変更理由]　　名称変更
住　所　　愛知県刈谷市昭和町１丁目１番地
氏　名　　株式会社デンソー